MongoDB vs DocumentDB

Cosmin Pintoiu
Solution Architect at Bigstep

@BigStepInc

bigstep.com

Cosmin Pintoiu

Solution Architect at Bigstep

Designed and implemented critical message processing projects in financial sector and real-time analytics in retail sector.

Currently focused on large-scale real-time implementations, Data lakes and machine learning using Tensorflow.

Agenda:

- Intro to Mongo and DocumentDB
- Setup Methodology
  - Node types
  - Network and AZ
- Benchmark using
  - Jmeter and custom sampler
  - YCSB
  - Mongo Socialite
- Price consideration
- Conclusions
- Q&A

Duration: 25m – 30m

MongoDB and DocumentDB

In this study, we will take a look at performance and cost aspects of running a MongoDB database environment on Bigstep Metal Cloud versus DocumentDB from AWS. To make it a fair comparison we will use similar resources and identical load tests.

- **MongoDB** is a cross platform document oriented database, was released 10 years ago and offers a multitude of features: indexing, replication, load balancing, aggregation, transactions.

- **Amazon DocumentDB** (with MongoDB compatibility) is a fast, scalable, highly available, and fully managed document database service that supports MongoDB workloads.

Our scope is to make this study impartial and easily reproducible, in this regard all the steps involved in setting up the environment and the test are available on github (https://github.com/ccpintoiu?tab=repositories).

Benchmarks

Considerations when performing benchmarks*:

- **Relevant** (for users of the benchmark: engineering, marketing, buyers etc)
- **Repeatable** (results)
- **Fairness** (to both hw and sw involved)
- **Verifiability** (in case of audit)
- **Economical** (to set up, run and publish)

* key aspects according to: Performance Evaluation and Benchmarking for the Era of Artificial Intelligence TPCTC 2018

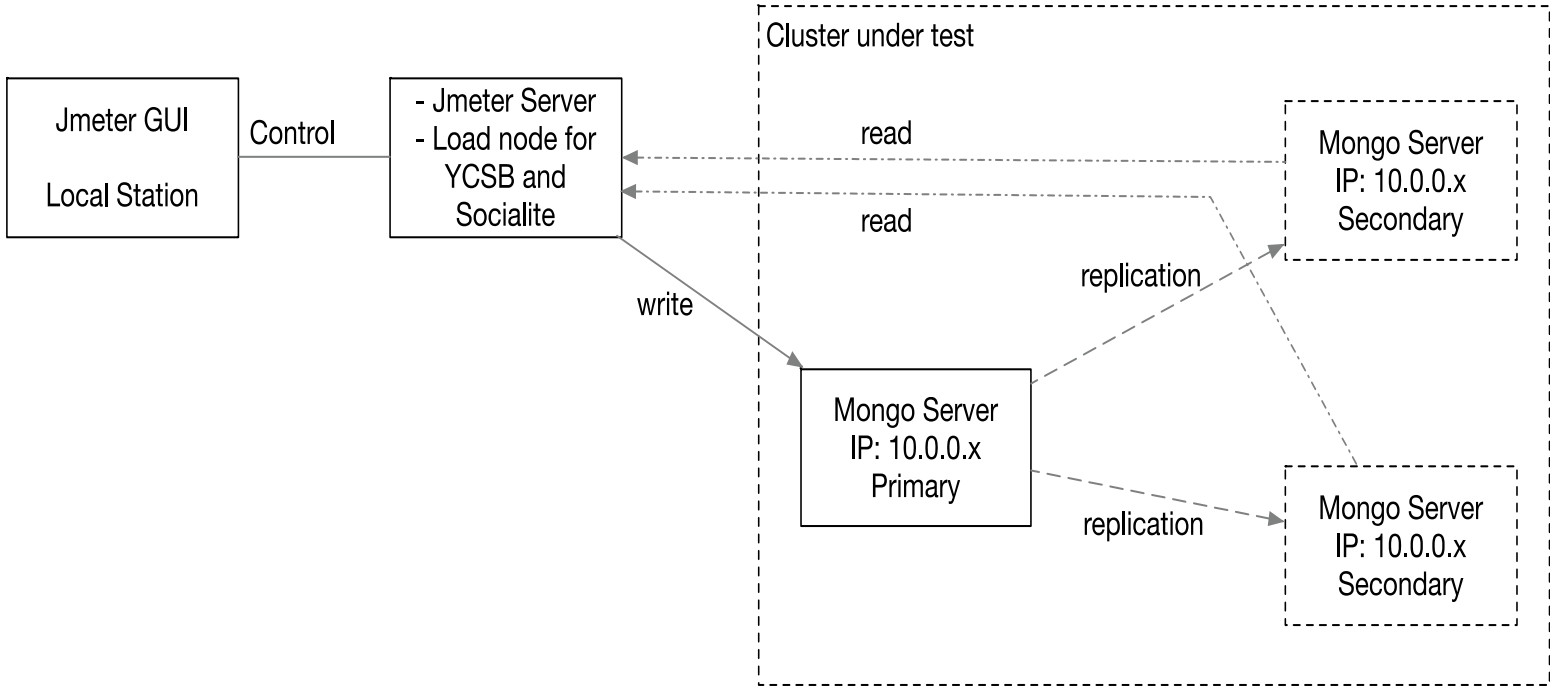
Authors: Raghunath Nambiar and Meikel Poess

eBook:

https://play.google.com/store/books/details?id=ps6FDwAAQBAJ&rdid=book-ps6FDwAAQBAJ&rdot=1&source=gbs_vpt_read&pcampaignid=books_booksearch_viewport

Node types

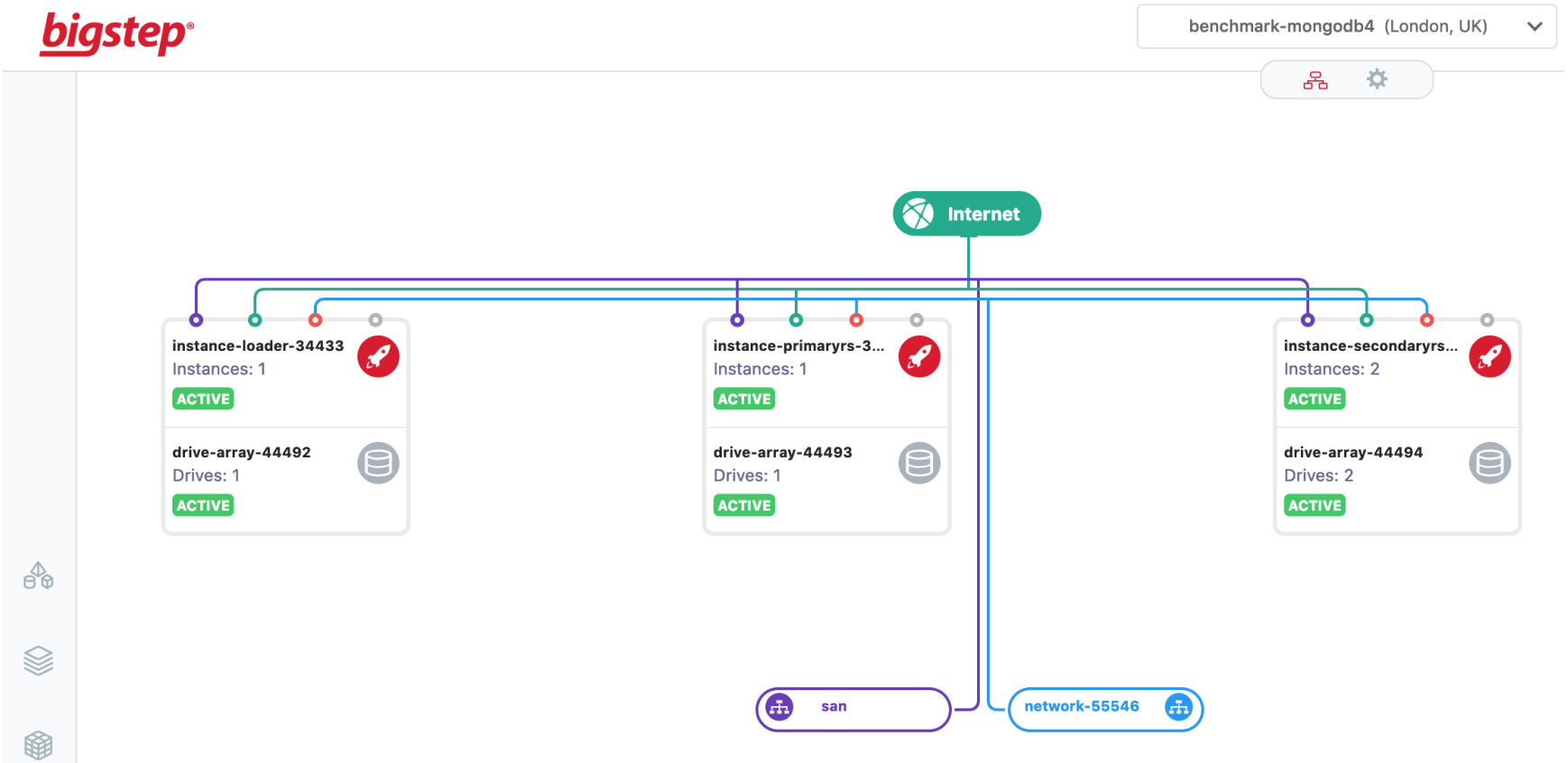
	Model	CPU	Memory	Storage	Network Performance
Bigstep MongoDB	FMCI 8.32	8*	32 GB ECC	BSA	4 x 10 gbps
AWS DocumentDB	db.r4.2xlarge	8 (vCPU)	61 GB	EBS-only**	high
AWS MongoDB EC2	m5.2x.large	8 (vCPU)	32 GB	EBS-only**	Up to 10 gb



Setup Mongo on Bigstep

- https://ctrl.bigstep.com/en/infrastructure/diagram?infrastructure_id=2887
- 1 Load node + 3 Mongo nodes
- Version 4.0.1

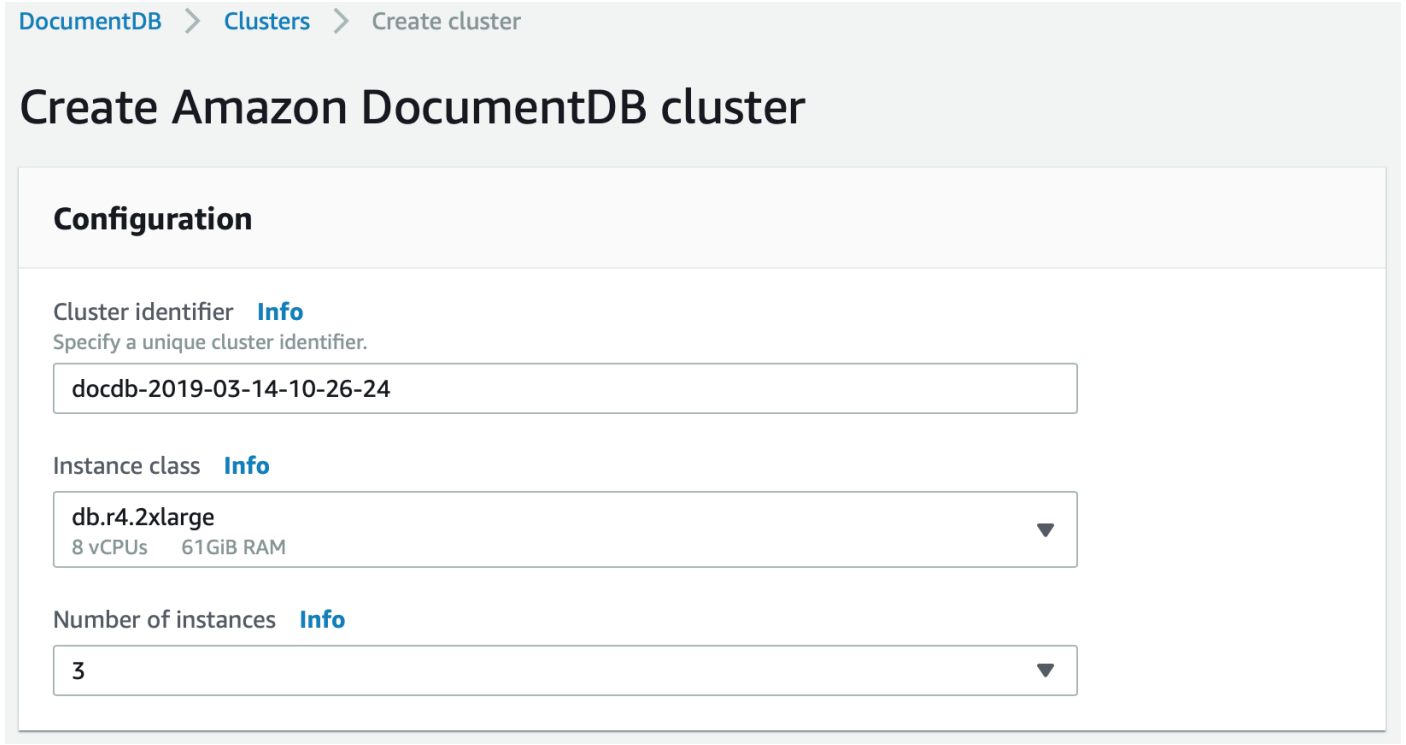
```
Install mongo DB cluster:

- set up private ips and /etc/hosts

- add repo:

- yum -y install mongodb-org

- create folder: mkdir /data/mongod

edit conf file on conf server: vim /etc/mongod.conf
add:  data file path and replSetName: "mongo_rs"

- start the server:

mongod --config /etc/mongod.conf

rs.initiate()

rs.add("mongod1")

rs.status()

rs.isMaster()
```



Setup DocumentDB

- 1 Load node (EC2) + 3 Mongo nodes (db.r4.2xlarge)
- API version 3.6

```
- setup EC2 instance

- ssh

- add repo for mongo shell

- sudo yum install -y mongodb-org-shell

- mongo --ssl --host docdb

NOTE:

- both docdb and ec2 instance should be
in the same security group

- Connect from ouside to Document DB
using port forwarding: nc -zv host port
```



Benchmark using

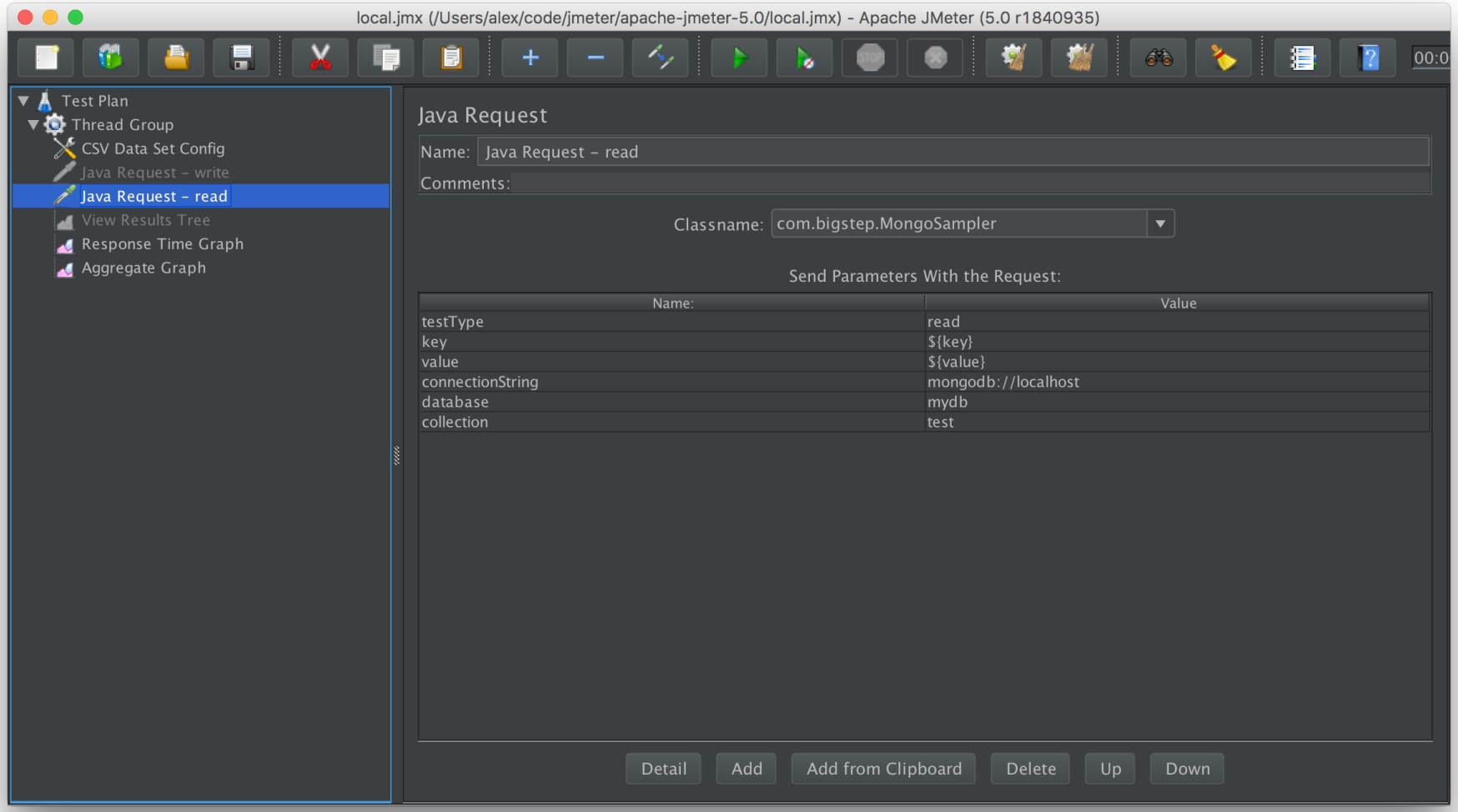
Jmeter is a load testing tool used mostly on web apps but it can be used very well on databases.
Is java based and supports variable parametrization.
This version uses ReactiveStreams 1.10 and the 3.9 Java MongoDB Driver and is tested with Jmeter version 5.0. It supports the following operations: read/write and readMany/writeMany.

YCSB is popular tool when comparing relative performance on NoSQL databases. Developed at Yahoo! for the specific purpose of comparative studies of various databases systems, YCSB is highly customizable. Workload files with 50/50 reads/writes, 75/25 and 95/5 to have a valid comparison.

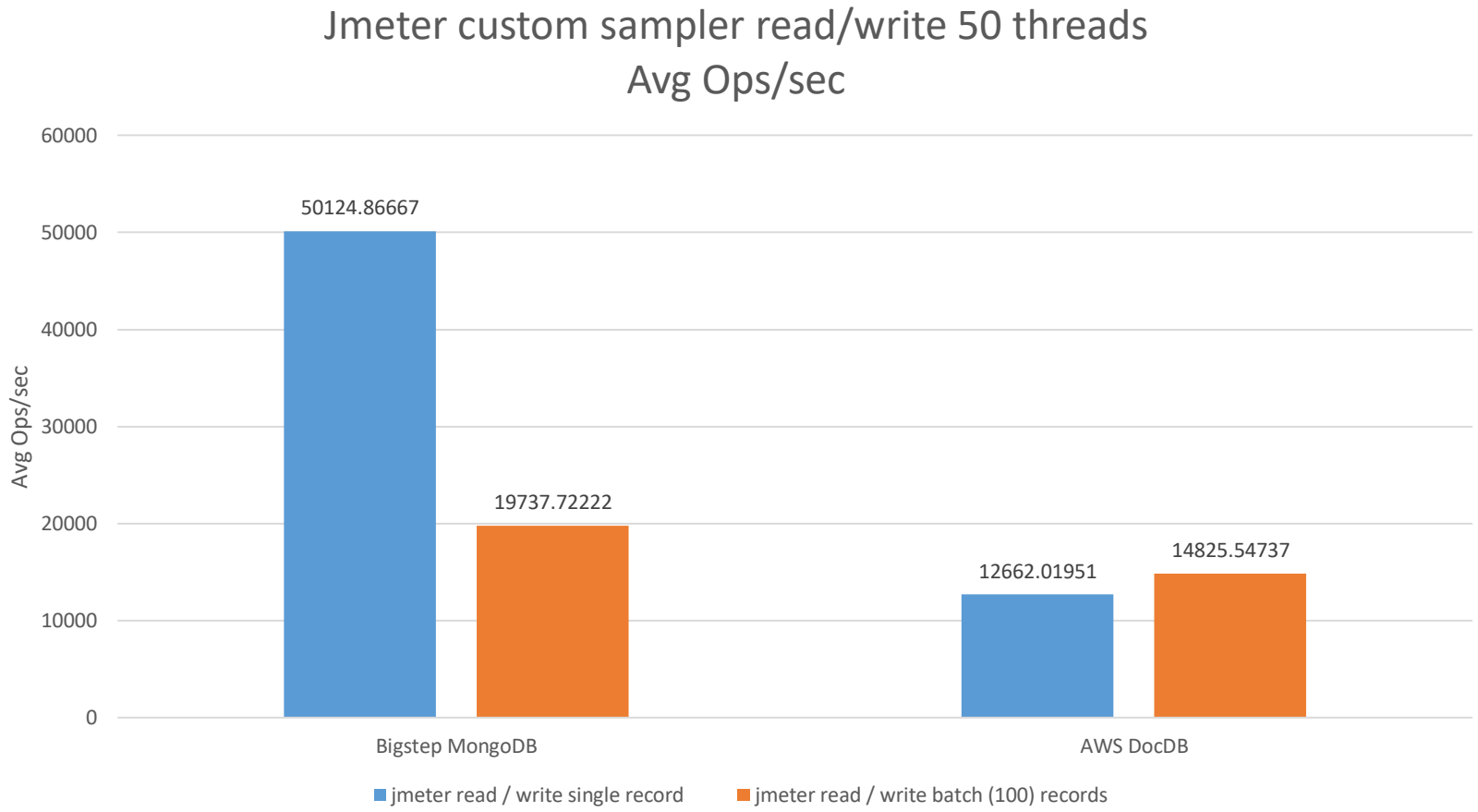
Socialite is a test developed by Mongo team part of their regression testing for mongo product.
This test simulates a social media platform with a number of users, followers and articles per user.
The run command reads the first 100 iterations and writes the results into a file. The output file is quite rich, most important field is the mean_rate, which shows the average ops/sec.

Jmeter test

https://github.com/bigstepinc/jmeter-mongo-db-custom-sampler
https://github.com/bigstepinc/jmeter-mongo-db-custom-sampler/releases/latest



Jmeter test



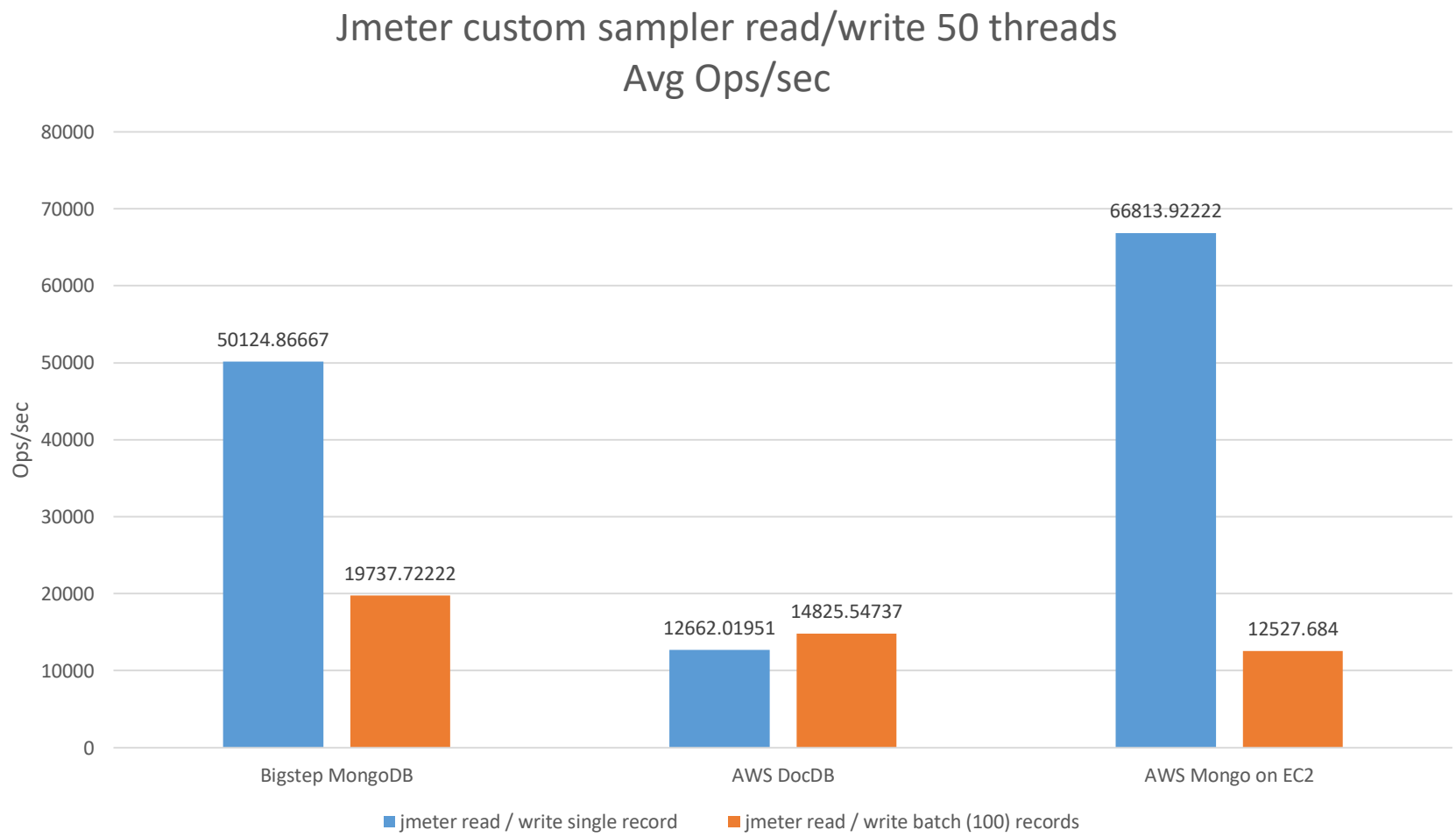
Jmeter config file used:

50 threads (simulates users)
loop count: 40000 (how many times a thread group gets executed)

Run command and time:
./jmeter.sh -n -t /tmp/Jmeter-Bigstep_1.3_WRSingle4M.jmx -l /tmp/output_jmxWRSingle4M.csv

Jmeter test

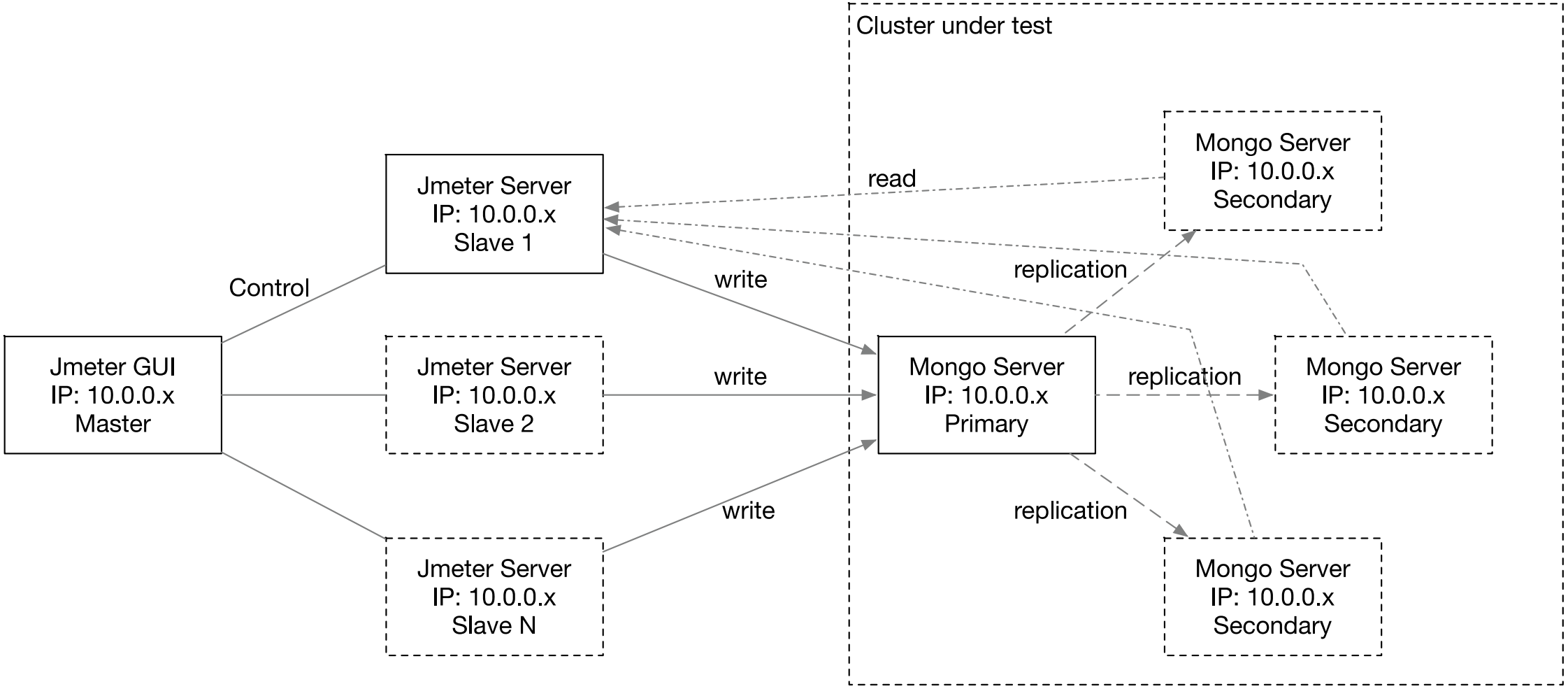
3rd test including Mongo Db on AWS EC instance (all instances in one availability zone)



Jmeter distributed test

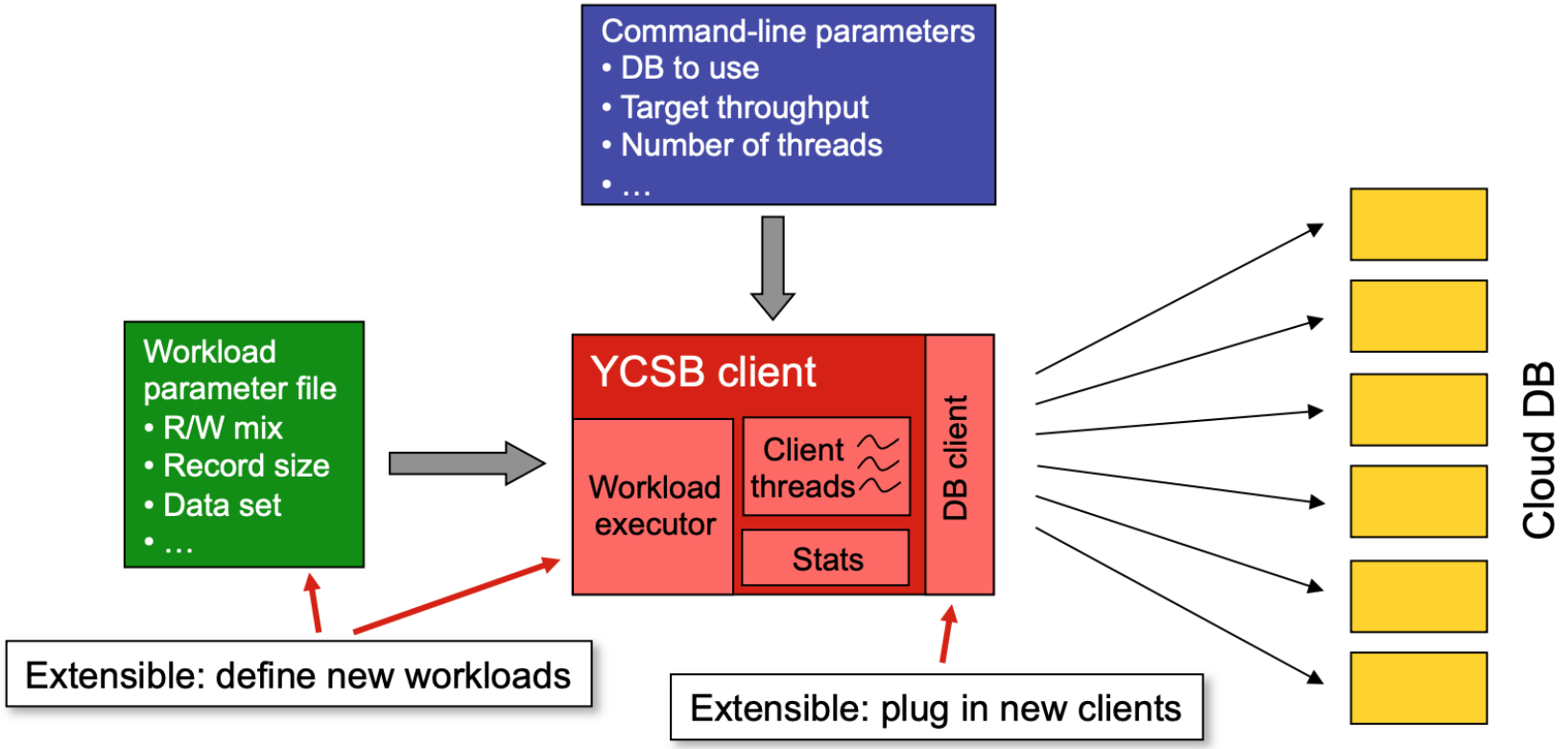
Next steps: Stress test using Jmeter Distributed testing

We can use our Custom Mongo Sampler (one other option is: https://github.com/johnlpage/POCDriver)



YCSB test

The goal of YCSB project is to develop a framework and common set of workloads for evaluating the performance of different "key-value" and "cloud" serving stores.



https://s.yimg.com/ge/labs/v1/files/ycsb-v4.pdf Author: Brian F. Cooper

YCSB test

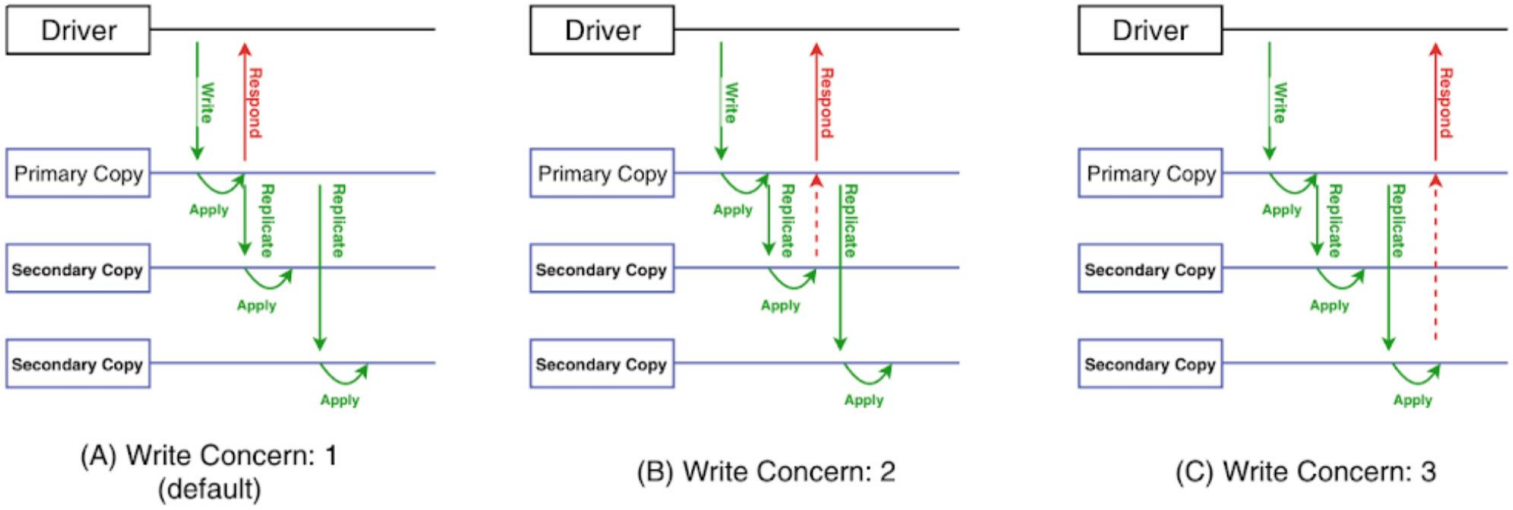
Load command and time:

./bin/ycsb load mongodb -s -P workloads/workload_small -threads 32 -p mongodb.url=mongodb://10.0.0.31:27017/?replicaSet=mongo_rs&w=majority

Load time:	Bigstep MongoDB	AWS DocDB
load 4M	9m44.324s	45m44.28s

Example workload file used:

requestdistribution=zipfian
recordcount=4096000
operationcount=20000000
readallfields=true
readproportion=0.5
updateproportion=0.5



eBook: Performance Evaluation and Benchmarking for the Era of Artificial

YCSB test

- db.usertable.count()
  4096000
- 20GB
- W=majority
- 32 threads

writeConcern

•Allowed values are:

- errors_ignored
- unacknowledged
- acknowledged
- journaled
- replica_acknowledged
- majority

readPreference:

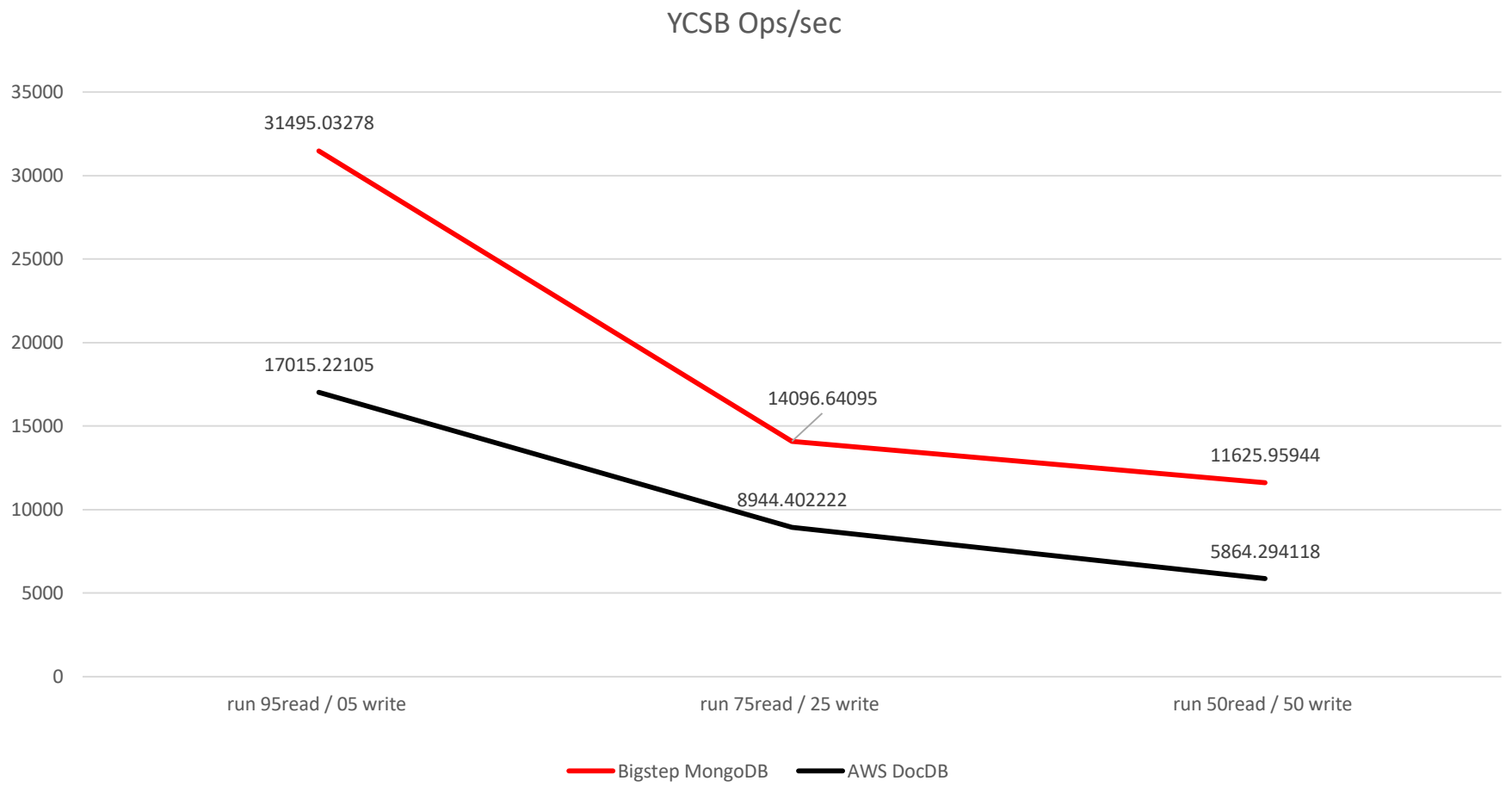
•Allowed values are :

- primary
- primary_preferred
- secondary
- secondary_preferred
- nearest

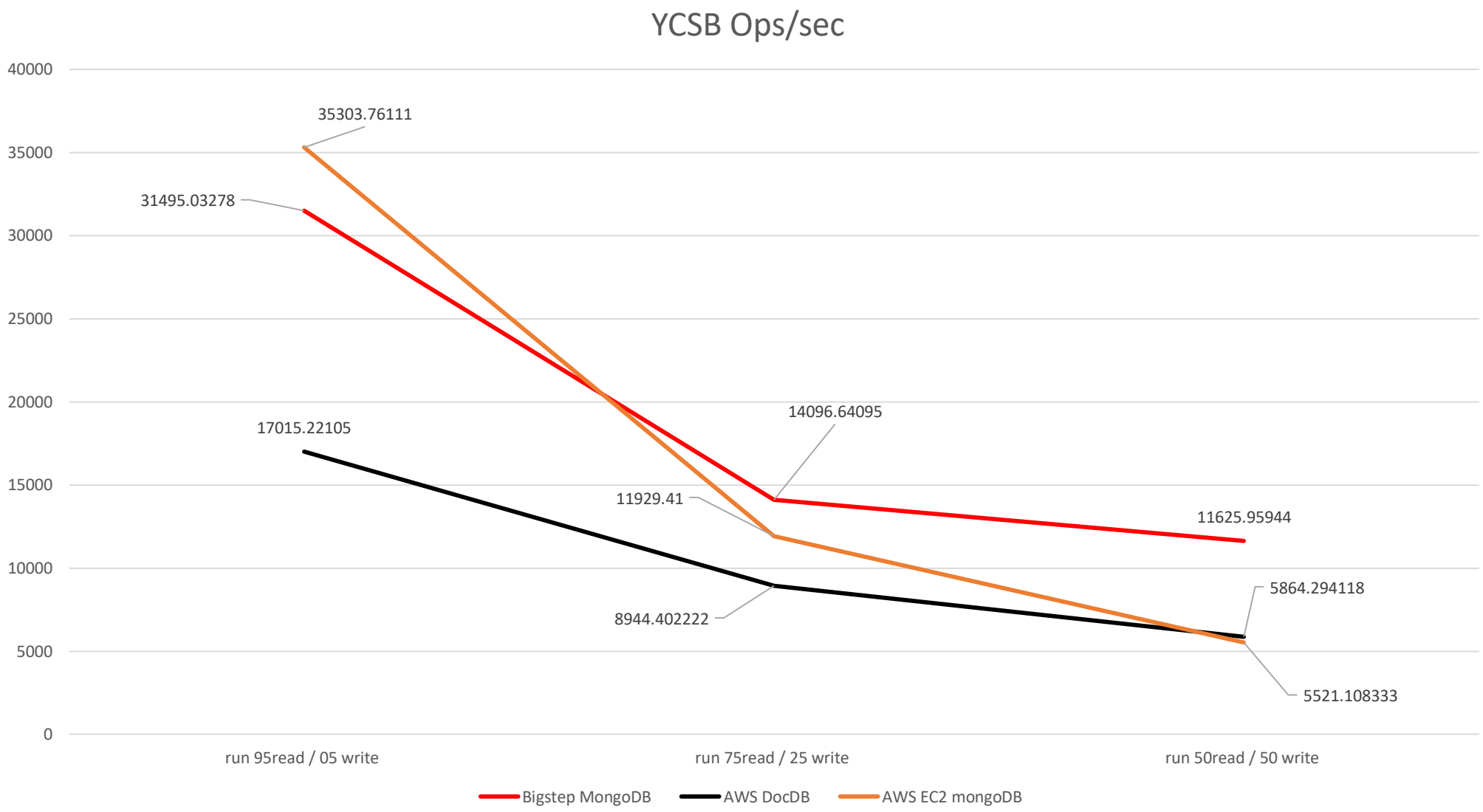

```
{
        "_id" : "user3801616714551441941",
        "field19" : BinData(0,"Pyc+MjA/KTY3JCU/OC0yKSM8KzYpKDAtJi88LSk6OzklODcvPTU4PzAhJCc9NCwnKCowJyM9NTIxKyszKCI6MzIsKi4yKT8pJzkzJiozJDg0NiggJCQuODMuOictLz4yKjE4NA=="),
        "field17" : BinData(0,"JywgID05LSsuND0vKiwmNDQpJzIoIiwoLTA0Li41JCU/PiohLDs5NTE5PjUkLDs5OywpNSYuISQzNzUpNCcpJSk3KDotJDs+MykzMzgkLjM9NCUkNyQjKz0sMCQ2OyEjMjk5JQ=="),
        "field18" : BinData(0,"MjM5JjY5NDcwJig1Ij8mMT4lOTcrICc3LCA6Myg2OCg6OicoITE4PSI8MDYoITc3IiIoLDQkPTcjPTAlPi8oIzYzJiApNTEjOCclJz0nNDIyPjInJissKj09PCo9Ky8uJSw1IA=="),
        "field22" : BinData(0,"JCw9PjM9PDEsPT4wIj0uOSo3IyQmOzI5MTc2KzY6KDIlPjYmIiU5MjclJjc4NyctISQ0MSI+LjM7Kz09PSEmIyU9LyIjNCI8OzohLS88Iik+KiwoND4lOSYwIiUtPzAsMT8kOA=="),
        "field23" : BinData(0,"OyApKT8gKT8rKywnNTY/ITkiIDI8LDEmJio9JT0jJSAoIyIqKygpJTcjNjIiMiklPTcjOzwwOyYnMCIkIS8uJT0/NCg+MiUqODswJyYwLSU0ITksJjgqNy05NSkoMTIpIzw6OQ=="),
        "field20" : BinData(0,"Li8/Kjc/MCg2OCA9LiQyOi0uMSM9MC02KCEqIi01LiQ2MiQgNywxKSkvMCo2OCQxPzg5PDQhOSIsMTkwKSkgJSEtPTo+KjktNyEyKiAkLykqNSo6KSQ0Pj4pOi0jISo0MjA4Ng=="),
        "field21" : BinData(0,"ITQgLzY1JD06Kz0lPiIiMz4zPTM/PSs2KyQoJi45PDwgNCEsLDI1NTs+LDQ/ICYmJT0hKjk7IywvNywjJzgsKScnIig3NiE/OSYuKDw4JSslIyQgJDo6MC4qNzc8JDkyND0wNw=="),
        "field24" : BinData(0,"Kyw7PSIvISc9NCUjNDU4MiE9ISsoLDUqPi8zOy8nNTAkMzcjISczPjc0MycrNTslKSImLTghKT88MDwvIT8sOi4/ISgzMyA0IzwsMToyJz8jNS03PSssOTsnNzMoMy49LCQ9Jg=="),
        "field11" : BinData(0,"MCgnODk3PzU4IzUrKyU0PDo0NjQxPTs5MiA7MC4mIDErOyQ7LjUsLC8xJz81LjA7OjUjIjMvJT8xNzErJC8kJzQlPDYzNCUrJzUgNyIgJSYmJCMmNy4sJTY1MDkyMTUxIDo0JA=="),
        "field12" : BinData(0,"PicjNjgyJSk3JCkiPScjLiQ8JDEpPC4iLDs7IikoNjwwOSw6LjAqITQ5KSQ4OCkiITwjMzMgNTc5KjAjKiMsLiwgITQqOTA3PCArOyAiOjQpPSIjPjMmKSQ6MCM/KSQrPyctNg=="),
        "field1" : BinData(0,"MjohMD49LjgoPCY7OCM+NCoiICUtMTslIzswOCAtKSQzNj0wOik2LC8rPDYoKyozMSg3PTwnIjMzMSIzMzYtMDghISskJTAqPyY0MSU7JjYpNjE2NTkvOCIyIDg+PDsqKSE4Pw=="),
        "field0" : BinData(0,"Pz40MSwrLC82JzUrIik3JCgoNys/LC0jID06Kig1MzQ3MSM2MTknNygnNTAqOzc2Ij46Pz8iKjQvMiU1NyEjNisnIicmPSMnKTAhLyotJS8lLjklIzYwLjYmKDMqNjs5KyY7NA=="),
        "field10" : BinData(0,"Jj01KCkqMzgvNyAsLTElIy4oMj04LCkvLykiISo+Lzs4Ny0lKCo1ISQjICYmNikuMScwJDA4MjYoJy0yNjEoITo2MywiNTYqKSM3OjwiMC4sKyUuLi8oMjclKic+Jzk+LSooJA=="),
        "field15" : BinData(0,"MiM/MTsrLSEjKTMhMSIwNCc9KTogLSwwPzMvJyo0KCI5PD05MDs7PzYhKCAiLT0wKiEwOS40IjwzKj06PSghND8uKCg3OSgqNTMqPi87MSAsLSwvOSg1PzghKzw2MCQ5ID0/Lg=="),
        "field16" : BinData(0,"LDs/IS81NyMjNzciJCooPzQ8PjI2NSs8JjgqNiM9LCgqJiYnJCsuPSooKy86JD0uNzkrMiw6LjszIDokLiAwPSY+OzU4KCkkIyE3IDImPTozMyIhPSUgNTksIjs3PTo2Ijo+MA=="),
        "field13" : BinData(0,"KT0vIjE4JSM4JyEnLjM8PSYvPD47NCM1JiEoNSgzMSo8NDEvIDE9PTAlLzghPyw5OyA0PjUiNyskJyE1PzM+NCIuPjk5NCwlNjEgPTUnJjcmPSMhODkjJyUvPi4uIj4iJzA4KQ=="),
        "field14" : BinData(0,"NyAuLyQgKzw3JCMuMSMnNzUnICktLjI4Iz0vIyE3NiI3NTIrLyshIyY1Ni83JzMuLyk/JCo2KysgPSUgIS4vKz0pOSAyJCs0Ii4iPzs+OjYkPS4pICU8IyInMjw/OiQgOSIyPw=="),
        "field7" : BinData(0,"MTgwKT0pIyc9JCY9LjwgOzE7LiwzPSYnITQkJCUkOC43MTsiKiU6OiUuNjQjJSEwISwhMycyNyoxLD8/MDs2NS4hMDsiMSoxMyM1KzEtMzgoLz0qIDkxOS0uOiI6PSAgJDE+Lg=="),
        "field6" : BinData(0,"MykxPz0wMiUpLiE/JzQqIiI3OCMxNSoxIDEvPzc5OzAtPCkpLyAjLyE5JjoyNzY7ID02Lz86Ly4sNyw0JzkmJj8pISw8MDIyJDk+MCYiJzQoOSo4MSQmIDM4NyQ1PSspPTozNg=="),
        "field9" : BinData(0,"OzU4KCg3OSwtPTc6KS84IiwkKTI1ISU8MDUzPCQ8NDMvJyE2KSI+KSMuNyM7IT43ICwlNSsyISg8ITUzJC4gPis5PioxKz4tPyw+NjQ8JDstJT89JTosLS89KzUzPzsgOSA5KA=="),
        "field8" : BinData(0,"PT81KyE0MCwxJTY3ITY9OzI2ODQ4NCU/LzY4MyMmIC8nNSk6IyAqMjw6LjooJysgJyw4PyQ6PyE3NzArMDglKyU3JjYpKjIrMz0nPDcrPjAsKS4nNDsxLjE/NCYrKjwpIz4jKg=="),
        "field3" : BinData(0,"MzkiKjs5MyMkMi0xNyc8Oy0zPy8jPi4yNzoxNzwiNTgmIj09PyY0Oj4vOjU3KSk3MTUrJjEqJSwrLTA8Lzk1MyEsJz42PiY0NCAhPC8rOzw1MS8rJDQrLio0LCkhJC4jJi02MQ=="),
        "field2" : BinData(0,"NyU1NSI9NiUwICEwNSg6IisqIT4iMzY+MDYtKDMzPTIhOyA7KzglJy0lPTAhIjwyLDYrIDQ3NjQyJCMvJTMkMC88LjktIyQgMTAhMSwtJzg5JyMxPj89LzcgKi41PichOyonPA=="),
        "field5" : BinData(0,"PyAvNCk+KDkyLDMsLig3JTQlMCYhNCkkIi06MSwxPy4nJyA0IjQ4LyAvMSogPyI9JDsiKiQ3ISIiLzk/OzMqLTcxMjgkMTQyNDI/JjEkPTkoKzYkIDYnISQ+LCs0Nys5PC08Lw=="),
        "field4" : BinData(0,"KjQ+NTsuMDsiNj84MT8rJywyJiYyOikxOy4qOD49IjExPC4qICYgOSg5ISIkKDcjMComNzIpLjooOygmOyAyPz4gJSUjODEuLykjJTI0Ki83PSouNiktMSU1NCIpNCEgMyY/PQ==")
}
```

YCSB test

The first step is to load **4 million records** using 32 threads and count the time that each environment needs to complete the task.



YCSB test


YCSB Ops/sec
40000
35000
30000
25000
20000
15000
10000
5000
0
35303.76111
31495.03278
17015.22105
14096.64095
11929.41
11625.95944
8944.402222
5864.294118
5521.108333
run 95read / 05 write
run 75read / 25 write
run 50read / 50 write
Bigstep MongoDB
AWS DocDB
AWS EC2 mongoDB

Socialite

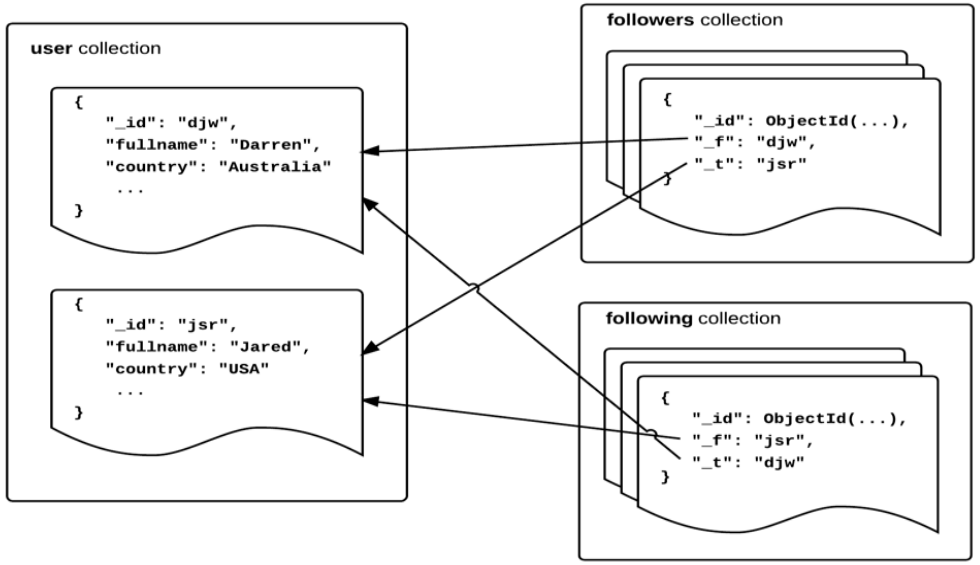
As with the YCSB tool, Socialite is quite complex and offers various load tests: benchmark, timeline-read-follower-ramp, send-ramp-followers.
The Socialite implementation uses 3 MongoDB collections by default called users, followers, following.

Load command and time:
java -jar ./target/socialite-0.0.1-SNAPSHOT.jar load --users 100000 --maxfollows 5000 --messages 20 --threads 32 sample-config.yml

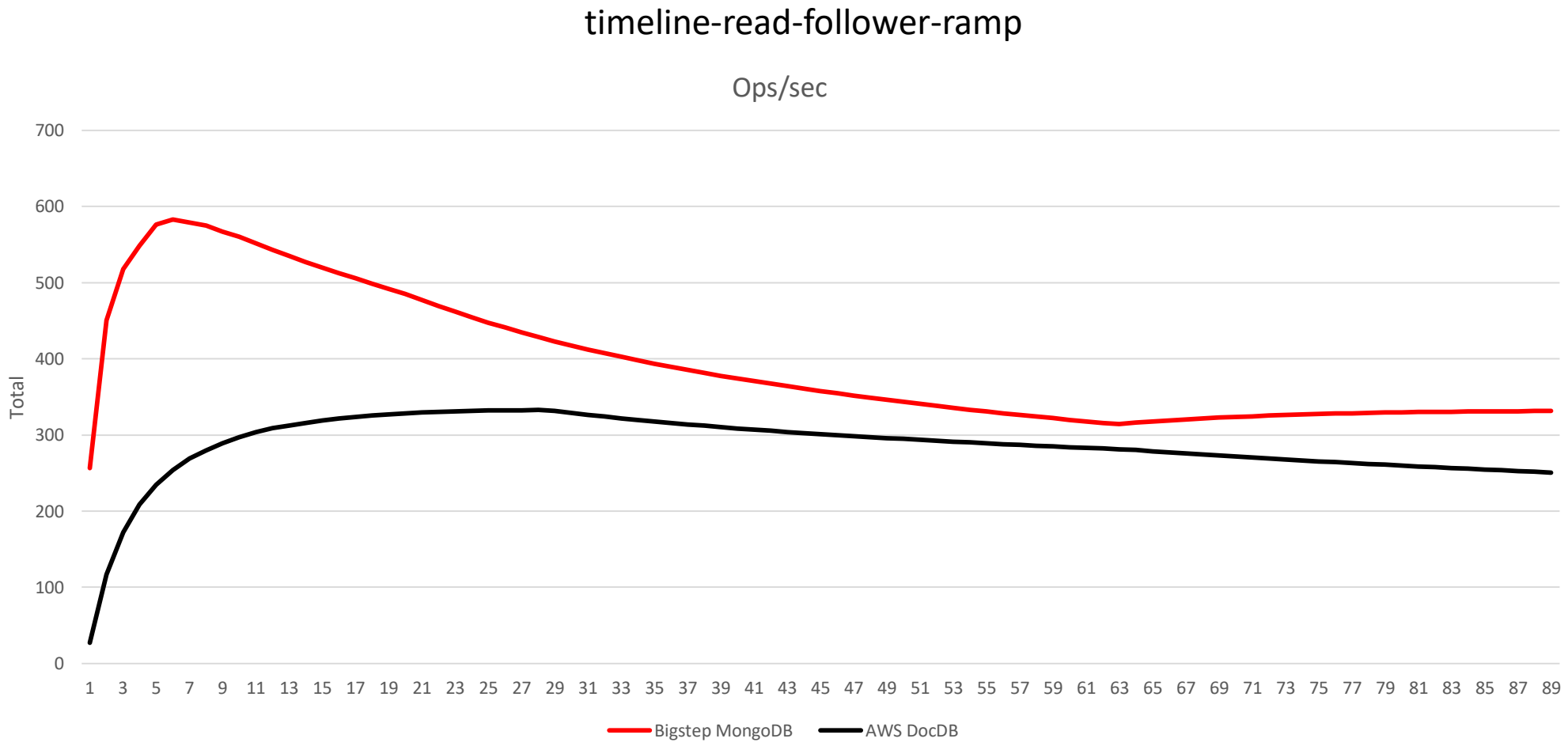
Run command:
java -jar ./target/socialite-0.0.1-SNAPSHOT.jar timeline-read-follower-ramp --out output1 --start 1 --stop 100 sample-config.yml

Example config file used:

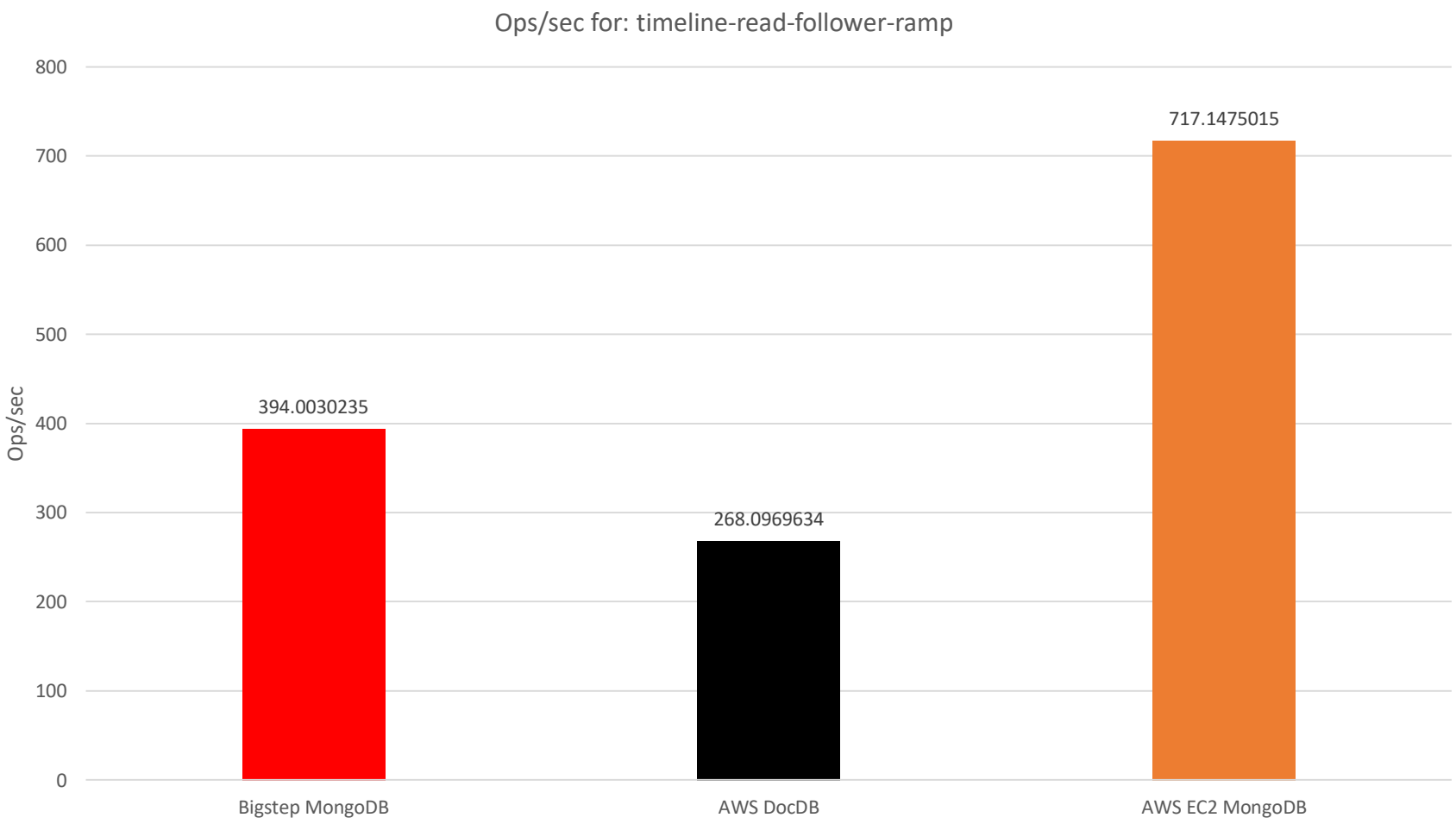
totalUsers=10000
activeUsers=1000
duration=3600
sessionDuration=30
concurrency=512
maxFollows=5000
messages=20



Socialite


timeline-read-follower-ramp
Ops/sec
Total
700
600
500
400
300
200
100
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89
Bigstep MongoDB
AWS DocDB

Socialite


Ops/sec for: timeline-read-follower-ramp
Ops/sec
800
700
600
500
400
300
200
100
0
394.0030235
268.0969634
717.1475015
Bigstep MongoDB
AWS DocDB
AWS EC2 MongoDB

Cluster deployment and scalability

For the moment **Bigstep** does not provide managed services on MongoDB. You can install the software off the shelf on the bare metal instances. The main advantage is that you can configure the cluster accordingly to your needs. Being a self-manage platform you can also setup up a Sharded Cluster which will offer you better performance on large amount of data.

AWS **DocumentDB** is ready for production and you can start loading data as soon as the instances are up. You don't have access on the management side of the services so the disadvantage is that you cannot customize. You use as it is.

	Create Cluster	Install Mongo	Scale (1 node)	Attach Workers	Total time:
Bigstep Platform	12 min	10 min	-	-	22 min
AWS DocDB	12 min	-	4 min	2 min	18 min

Cluster deployment and scalability

Steps to take for scalability Bigstep or AWS EC cluster:

- deploy new instance
- configure private IP
- install mongodb
- add the new node in cluster

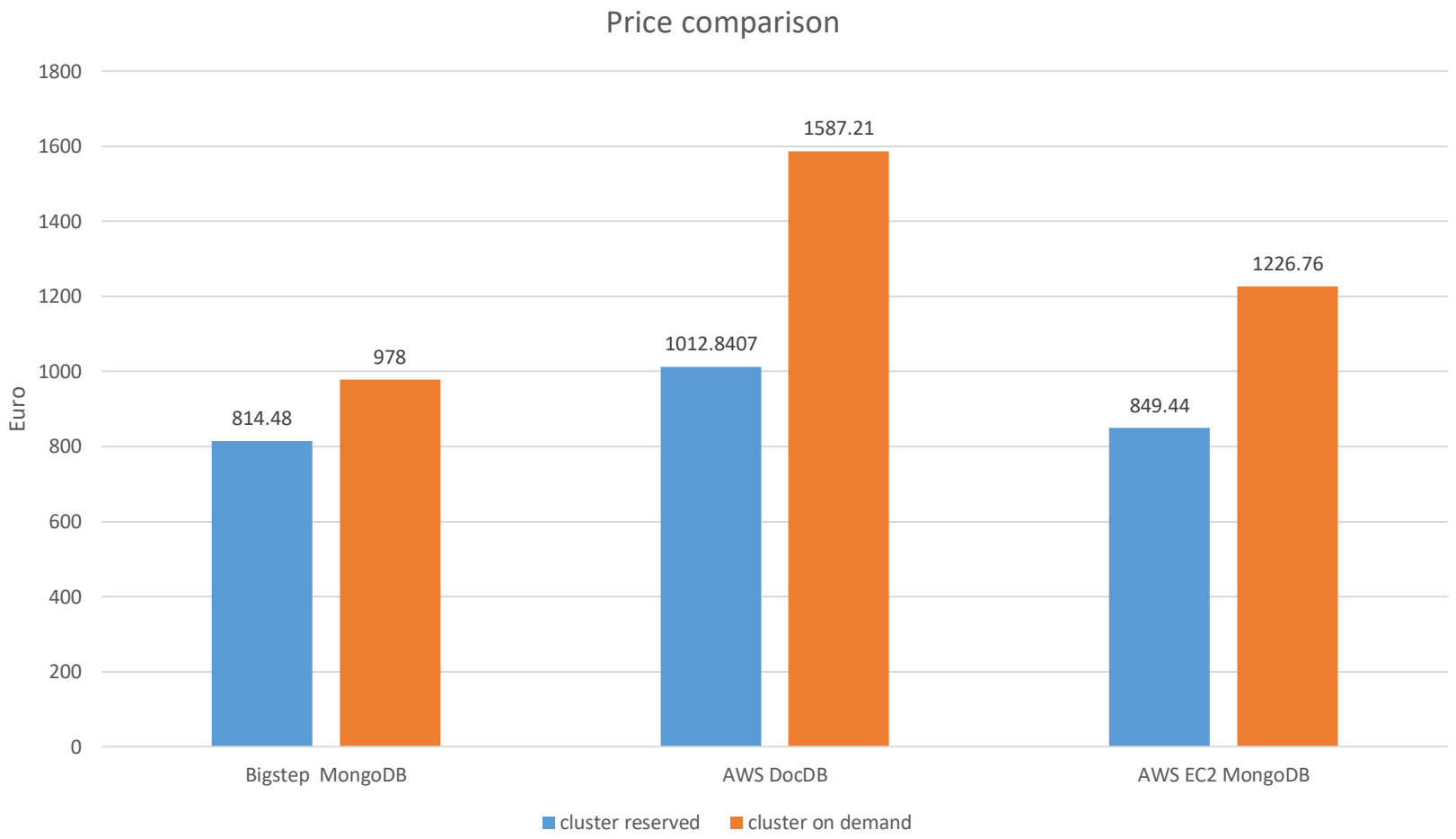
The work time is higher than DocumentDB as at the moment Bigstep does not provide MongoDB as an integrated service, yet some steps can be automated. DocumentDB is way easier to scale up to 15 replica nodes and grows the size of your storage volume automatically

Scale Cluster	Add Instance	Conf IP	Install Mongo	Add node in cluster	Total time:
Bigstep Platform	5 min	2 min	5 min	5 min	17 min
AWS DocDB	5 min	-	-	-	5 min

Costs for entire cluster:

FULL METAL CLOUD INSTANCES	Description	Quantity	Reserved price (EUR/mth)	On demand price (EUR/h)	Reserved price (USD/mth)	On demand price (USD/h)	Total Monthly Reserved price (EUR)	Total Hourly On demand price (EUR)
FMCI 8.32v2 - General Purpose	CPU: 1 x Intel Xeon E3-1585L v5 - 4 cores 8 threads @ 3.3Ghz RAM: 32 GB DDR4 ECC Network: 2 x 10 GbE	1	189	0.30			€ 189.00	€ 225.00
Storage	SSD Storage (100 GB)	100	0.15	0.00			€ 14.62	€ 19.50
Bigstep MongoDB							**€ 203.62**	**€ 244.50**
AWS r4.2xlarge	8 (vCPU) 61 GB EBS-only*	1	256.51	0.56	288.2	0.62	€ 256.51	€ 416.52
Storage	First 50 TB / Month - price per GB	100	0.02		$0.02		€ 10.32	€ 10.32
AWS 1							**€ 266.83**	**€ 426.84**
AWS m5.2x.large	8 (vCPU) 32 GB EBS-only*	1	202.06	0.40	227	0.44	€ 202.06	€ 296.37
Storage	First 50 TB / Month - price per GB	100	0.02		$0.02		€ 10.32	€ 10.32
AWS 2							**€ 212.38**	**€ 306.69**

Costs for entire cluster:



General Takeaways

Pick the right type of node

Perform custom tests for your problem

Take into account scalability and flexibility

I’m all ears!

@bigstepinc

cosmin.pintoiu@bigstep.com

bigstep®